B TECH
(SEM-III) THEORY EXAMINATION 2020-21
COMPUTER ORGANIZATION AND ARCHITECTURE

Section A

Ques 1. Define the term Computer architecture and Computer organization.

Ans: Computer Architecture is concerned with the way hardware components are connected to form a computer system. Computer Organization is concerned with the structure and behaviour of a computer system as seen by the user. It acts as the interface between hardware and software.

Ques 2. What is mean by bus arbitration? List different types of bus arbitration.

Ans: Bus arbitration is the process by which the next device to become the bus master is selected and bus mastership is transferred to it. The selection of bus master is usually done on the priority basis.

There are two approaches to bus arbitration: Centralized and distributed.

1. Centralized Arbitration

    a. Daisy chaining

    b. Polling method

    c. Independent request

2. Distributed Arbitration

Ques 3. Discuss biasing with reference to floating point representation.

Ans: In IEEE 754 floating-point numbers, the exponent is biased in the engineering sense of the word – the value stored is offset from the actual value by the exponent bias, also called a biased exponent.[1] Biasing is done because exponents have to be signed values in order to be able to represent both tiny and huge values, but two's complement, the usual representation for signed values, would make comparison harder.

To solve this problem the exponent is stored as an unsigned value which is suitable for comparison, and when being interpreted it is converted into an exponent within a signed range by subtracting the bias.

Ques 4. What is restoring method in division algorithm?

Ans: A division algorithm provides a quotient and a remainder when we divide two number. They are generally of two type slow algorithm and fast algorithm. Slow division algorithm are restoring, non-restoring. Restoring term is due to fact that value of register A is restored after each iteration.

Ques 5. Define micro operation and micro code.

Ans: Micro instructions are the instructions stored in control memory. In processor design, Microcode is a technique that interposes a layer of computer organization between the CPU hardware and the programmer-visible instruction set architecture of the computer.

Ques 6. Write short note on RISC.

Ans: A reduced instruction set computer (RISC) is a computer that uses a central processing unit (CPU) that implements the processor design principle of simplified instructions. To date, RISC is the most efficient CPU architecture technology.

With RISC, the basic concept is to have simple instructions that do less but execute very quickly to provide better performance.

Ques 7. Define hit ratio.

Ans: A hit ratio is a calculation of cache hits, and comparing them with how many total content requests were received.

Ques 8. What do you mean by page fault?

Ans: A page fault (sometimes called #PF, PF or hard fault) is a type of exception raised by computer hardware when a running program accesses a memory page that is not currently mapped by the memory management unit (MMU) into the virtual address space of a process.

Ques 9. Explain the term cycle stealing.

Ans: In computing, traditionally cycle stealing is a method of accessing computer memory (RAM) or bus without interfering with the CPU. It is similar to direct memory access (DMA) for allowing I/O controllers to read or write RAM without CPU intervention.

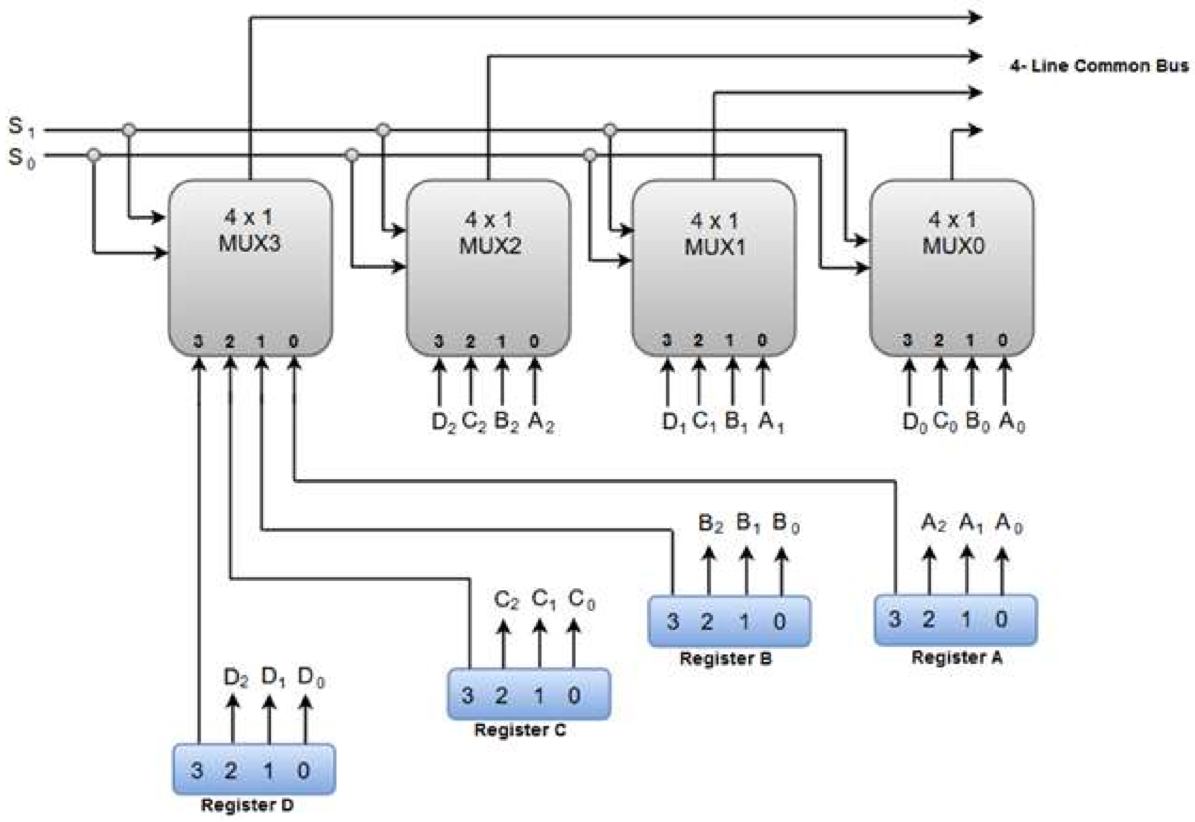
Ques 10. What do you mean by vector interrupt? Explain.

Ans: In a computer, a vectored interrupt is an I/O interrupt that tells the part of the computer that handles I/O interrupts at the hardware level that a request for attention from an I/O device has been received and and also identifies the device that sent the request.

Section B

Ques 1. (i)Draw a diagram of bus system using MUX which has four registers of size 4 bits each.

Bus System for 4 Registers:



Ans:

(ii)Evaluate the arithmetic statement.

x = A + B * [C * D + E * (F + G)]

using a stack organized computer with zero address operation instructions.

Ans:

Ques 2. Explain in detail the principle of carry look ahead adder and design 4-bit CLA adder.

Ans: **Motivation behind Carry Look-Ahead Adder :**

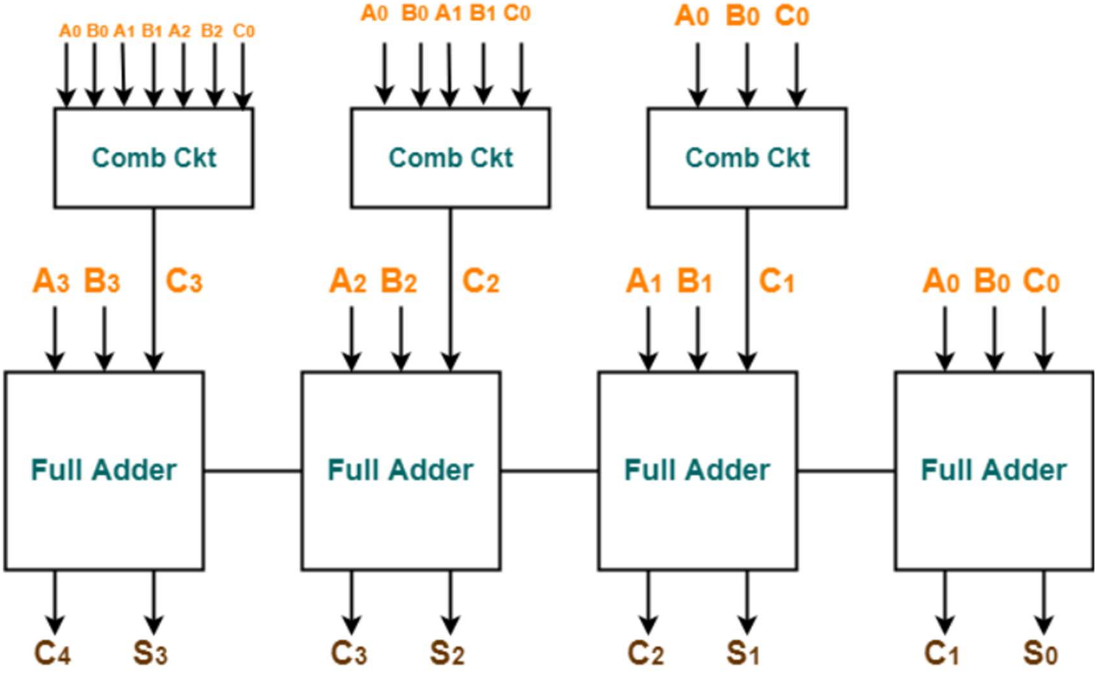
In ripple carry adders, for each adder block, the two bits that are to be added are available instantly. However, each adder block waits for the carry to arrive from its previous block. So, it is not possible to generate the sum and carry of any block until the input carry is known. The i^{th} block waits for the i-1^{th} block to produce its carry. So there will be a considerable time delay which is carry propagation delay.

Carry Look Ahead Adder-

Carry Look Ahead Adder is an improved version of the ripple carry adder.

It generates the carry-in of each full adder simultaneously without causing any delay.

The time complexity of carry look ahead adder = Θ (logn).



Carry Look Ahead Adder Logic Diagram

Ques 3. Draw the flowchart for instruction cycle with neat diagram and explain

Ans:

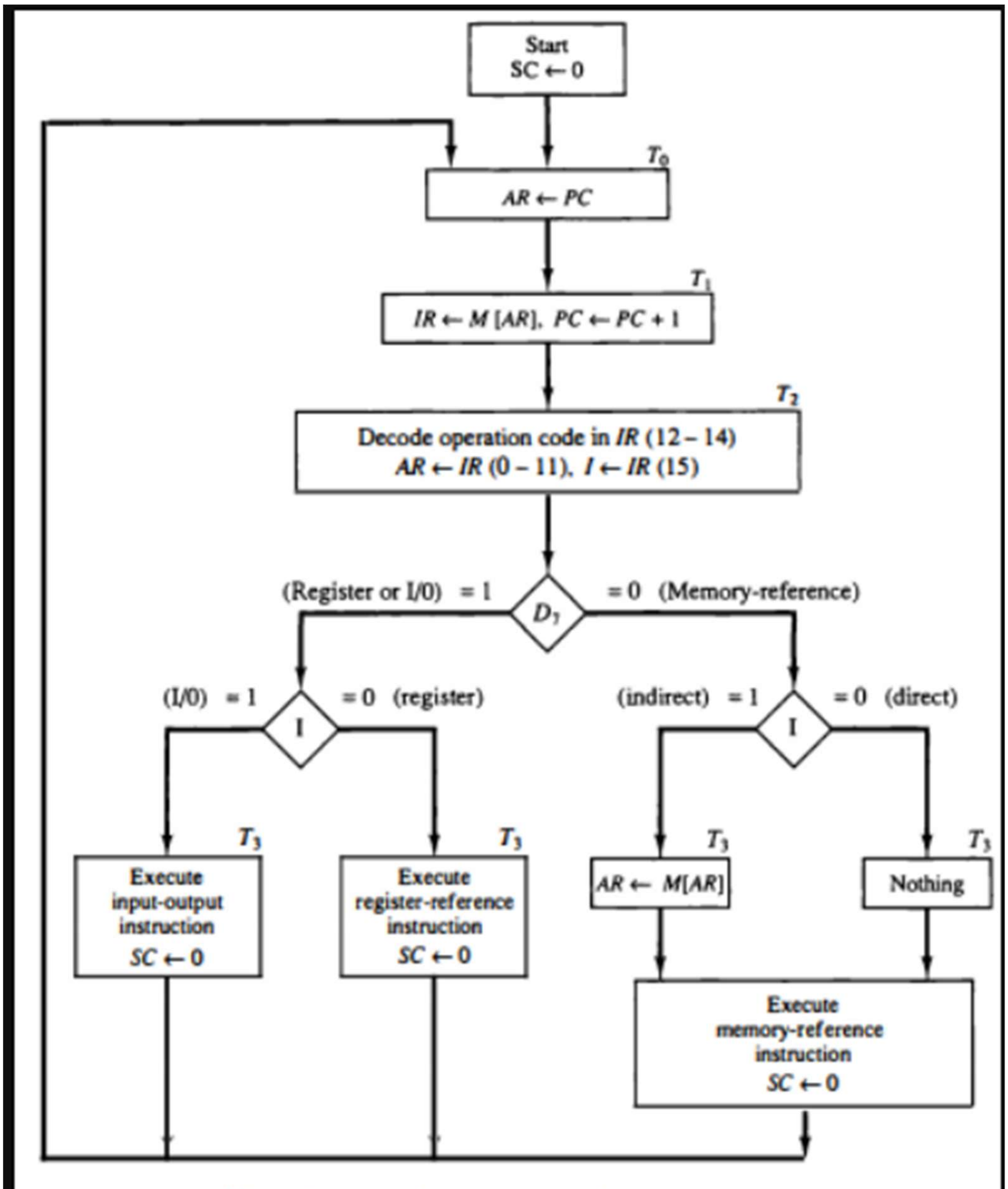

Start
SC ← 0
T_0
AR ← PC
T_1
IR ← M [AR], PC ← PC + 1
T_2
Decode operation code in IR (12 – 14)
AR ← IR (0 – 11), I ← IR (15)
(Register or I/0) = 1
D_7
= 0 (Memory-reference)
(I/0) = 1
I
= 0 (register)
(indirect) = 1
I
= 0 (direct)
T_3
Execute input-output instruction SC ← 0
T_3
Execute register-reference instruction SC ← 0
T_3
AR ← M[AR]
T_3
Nothing
Execute memory-reference instruction SC ← 0

Figure Flowchart for instruction cycle (initial configuration).

Ques 4. Discuss 2 D RAM and 2.5D RAM with suitable diagram.

Ans: The internal structure of Memory either RAM or ROM is made up of memory cells that contain a memory bit. A group of 8 bits makes a byte. The memory is in the form of a multidimensional array of rows and columns. In which, each cell stores a bit and a complete row contains a word. A memory simply can be divided into this below form.

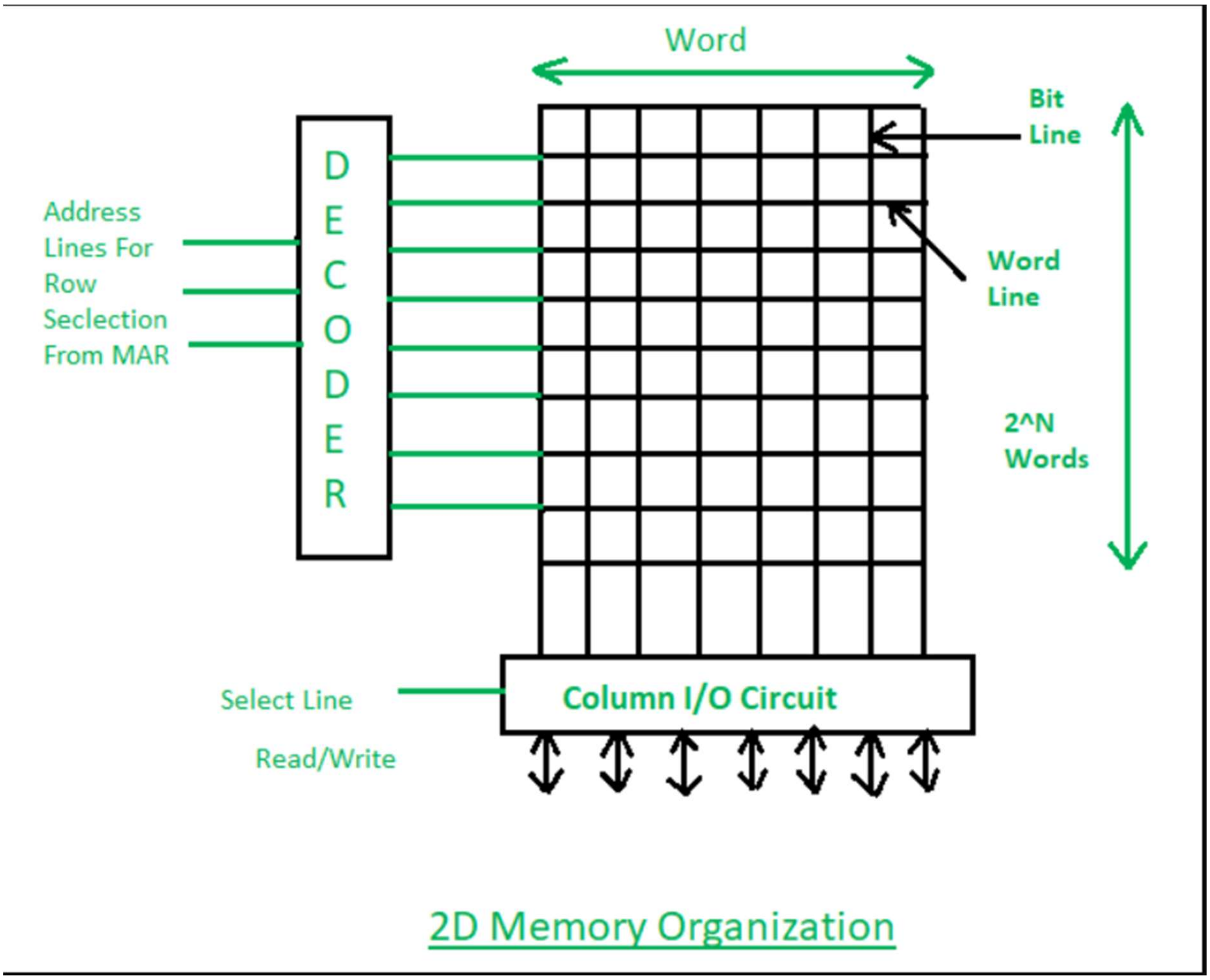
2^n = N

where,n is the no. of address lines and N is the total memory in bytes.

There will be 2^n words.

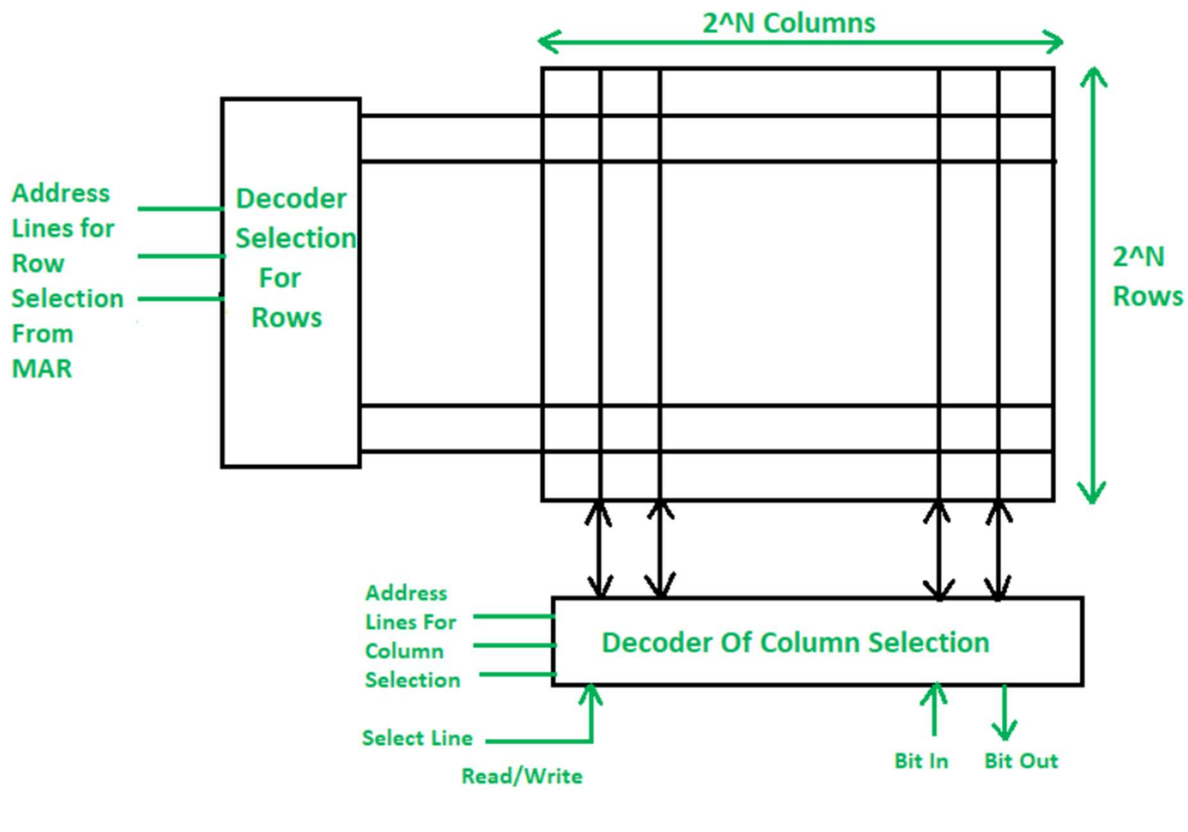
2D Memory organization –

In 2D organization, memory is divided in the form of rows and columns(Matrix). Each row contains a word, now in this memory organization, there is a decoder. A decoder is a combinational circuit that contains n input lines and 2n output lines. One of the output lines selects the row by the address contained in the MAR and the word which is represented by that row gets selected and is either read or written through the data lines.



2.5D Memory organization –

In 2.5D Organization the scenario is same but we have two different decoders one is column decoder and another is row decoder. Column decoder is used to select the column and row decoder is used to select the row. The address from the MAR goes as the decoders' input. Decoders will select the respective cell through the bit outline,then the data from that location will be read or through the bit in line data will be written at that memory location.


2^N Columns
Address Lines for Row Selection From MAR
Decoder Selection For Rows
2^N Rows
Address Lines For Column Selection
Decoder Of Column Selection
Select Line
Read/Write
Bit In
Bit Out

2.5D Memory Organization

Ques 5. Draw and explain the block diagram of typical DMA controller.

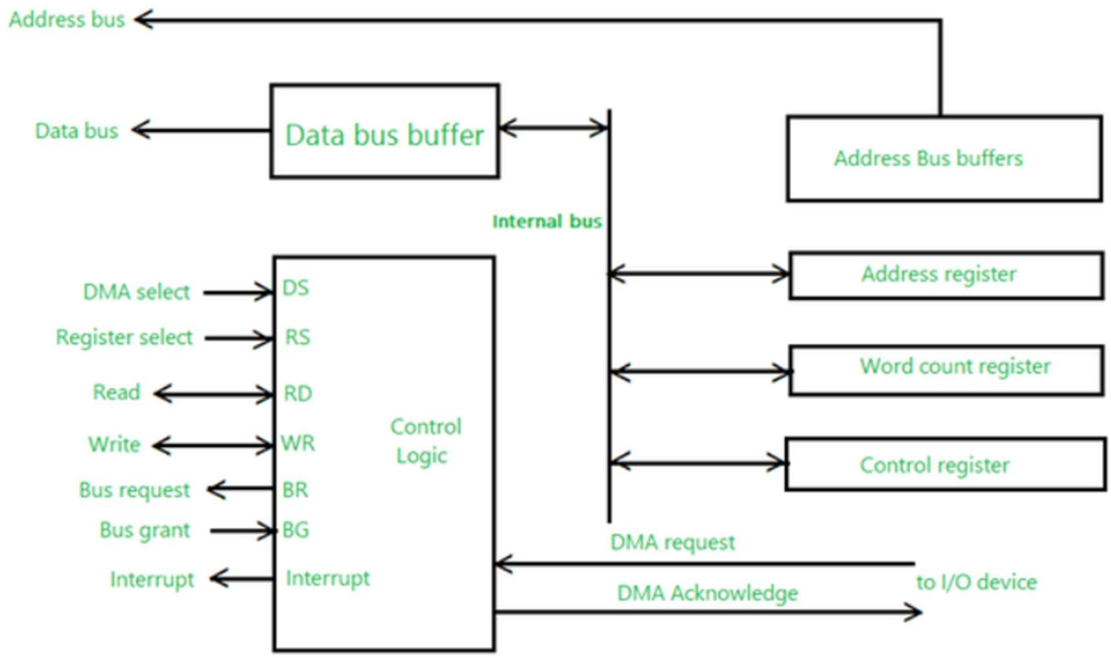
Ans: Direct Memory Access (DMA) :

DMA Controller is a hardware device that allows I/O devices to directly access memory with less participation of the processor. DMA controller needs the same old circuits of an interface to communicate with the CPU and Input/Output devices.

Fig below shows the block diagram of the DMA controller. The unit communicates with the CPU through data bus and control lines. Through the use of the address bus and allowing the DMA and RS register to select inputs, the register within the DMA is chosen by the CPU. RD and WR are two-way inputs. When BG (bus grant) input is 0, the CPU can communicate with DMA registers. When BG (bus grant) input is 1, the CPU has relinquished the buses and DMA can communicate directly with the memory.

DMA controller registers :
The DMA controller has three registers as follows.
Address register – It contains the address to specify the desired location in memory.
Word count register – It contains the number of words to be transferred.
Control register – It specifies the transfer mode.
Note – All registers in the DMA appear to the CPU as I/O interface registers. Therefore, the CPU can both read and write into the DMA registers under program control via the

data bus.



Explanation:

- The CPU initializes the DMA by sending the given information through the data bus.
- The starting address of the memory block where the data is available (to read) or where data are to be stored (to write).
- It also sends word count which is the number of words in the memory block to be read or write.
- Control to define the mode of transfer such as read or write.
- A control to begin the DMA transfer.

SECTION C

Ques 1. An instruction is stored at location 400 with its address field at location 401.The address field has the value 500.A processor register R1 contains the number 200. Evaluate the effective address if the addressing mode of the instruction is (i) direct (ii) immediate (iii) relative (iv) register indirect (v) index with R1 as index register.

Ans: Ques 1. An instruction is stored at location 400 with its address field at location 401.The address field has the value 500.A processor register R1 contains the number 200. Evaluate the effective address if the addressing mode of the instruction is (i) direct (ii) immediate (iii) relative (iv) register indirect (v) index with R1 as index register.

Ans:

Location _ Contents

400 _ opcode ;the instruction operation code

401 _ 500 ;address field of the above instruction

(a) direct addressing

Direct addressing means that the address field contains the address of memory location the instruction is supposed to work with (where an operand "resides").
Effective address would therefore be 500.

(b) immediate addressing

Immediate addressing means that the address field contains the operand itself.
Effective address would therefore be 401.

(c) relative addressing

Relative addressing means that the address field contains offset to be added to the program counter to address a memory location of the operand.
Effective address would therefore be 401 + 500 = 901.

(d) register indirect addressing

Register indirect addressing means that the address of an operand is in the register. The address field in this case contains just another operand.
Effective address would therefore be in R1 = 200.

(e) indexed addressing with R1 as index register

There are several possible indexed addressing modes but in this case (tehre is an address field) it is co called "indexed absolute" addressing.
In indexed absolute addressing the effective address is calculated by taking the contnts of the address field and adding the contents of the index register.
Effective address would therefore be 400 + R1 = 500 + 200 = 700.

Ques 2. What do you mean by processor organization? Explain various types of processor organization.

Ans:

A processor must have 3 functional units to be what we call a computer: a unit that performs arithmetic and logical operations on data (aka the ALU); a unit that remembers data while it isn’t being worked on (aka the Memory); and a unit which sequences the operations performed by the ALU and on which data (aka the Sequencer).

“Processor organization” is a term describing how those three elements are implemented and how they interconnect to accomplish their tasks.

In simple computer, the Sequencer consists of a Program Counter register (aka the PC), an incrementer for the PC, and a read/write port used for branches and subroutine calls. It also contains the Instruction Register which decodes instructions and drives the operation of the other two units.

The ALU consists of a literal ALU and an Accumulator register which is usually one operand and the destination for arithmetic and logical instructions. Control signals from the Sequencer indicate which operation is to be performed.

Memory is an array of storage slots indexed by an integer Placed in the Memory Address Register. The Memory Read/Write control signal indicates whether the value in the Memory Data Register should replace the contents of the addressed slot, or whether the contents of the slot should replace the contents of the Memory Data Register.

Single Accumulator based CPU organization.

The computers, present in the early days of computer history, had accumulator based CPUs. In this type of CPU organization, the accumulator register is used implicitly for processing all instructions of a program and store the results into the accumulator. The instruction format that is used by this CPU Organisation is One address field. Due to this the CPU is known as One Address Machine.

The main points about Single Accumulator based CPU Organisation are:

- In this CPU Organization, the first ALU operand is always stored into the Accumulator and the second operand is present either in Registers or in the Memory.
- Accumulator is the default address thus after data manipulation the results are stored into the accumulator.
- One address instruction is used in this type of organization.
- The format of instruction is: Opcode + Address
- Opcode indicates the type of operation to be performed.

Stack based CPU Organization :

The computers which use Stack-based CPU Organization are based on a data structure called stack. The stack is a list of data words. It uses Last In First Out (LIFO) access method which is the most popular access method in most of the CPU. A register is used to store the address of the topmost element of the stack which is known as Stack pointer (SP). In this organisation, ALU operations are performed on stack data. It means both the operands are always required on the stack. After manipulation, the result is placed in the stack.

The main two operations that are performed on the operators of the stack are Push and Pop.

General Register based CPU Organization

When we are using multiple general-purpose registers, instead of a single accumulator register, in the CPU Organization then this type of organization is known as General register-based CPU Organization. In this type of organization, the computer uses two or three address fields in their instruction format. Each address field may specify a general register or a memory word. If many CPU registers are available for heavily used variables and intermediate results, we can avoid memory references much of the time, thus vastly increasing program execution speed, and reducing program size.

For example:

MULT R1, R2, R3
This is an instruction of an arithmetic multiplication written in assembly language. It uses three address fields R1, R2, and R3. The meaning of this instruction is:

R1 <-- R2 * R3
This instruction also can be written using only two address fields as:

MULT R1, R2
In this instruction, the destination register is the same as one of the source registers. This means the operation

R1 <-- R1 * R2
The use of large number of registers results in short program with limited instructions.

Some examples of General register based CPU Organization are IBM 360 and PDP- 11.

The **advantages** of General register based CPU organization –

Efficiency of CPU increases as there are a large number of registers are used in this organization.
Less memory space is used to store the program since the instructions are written in compact way.

The **disadvantages** of General register based CPU organization –

Care should be taken to avoid unnecessary usage of registers. Thus, compilers need to be more intelligent in this aspect.
Since large number of registers are used, thus extra cost is required in this organization.

Ques 3. Show the systemic multiplication process of (20) X (-19) using Booth's algorithm

Ans:

Decimal validation:
20[010100]
×
13[001101]
= 260[000100000100]

Multiplier		Booth Multiplier
Bit[i]	Bit[i-1]	
0	0	0
0	1	+1
1	0	-1
1	1	0

Booth Multiplier Recoding Table:

Multiplier: 0 0 1 1 0 1 0

Booth Recoding (Click / Tap Me): 0 +1 0 -1 +1 -1

*A red zero is added after the least significant bit (LSB) for the conversion
Click on the zeros in "Booth Recoding" above to view the pair of bit of each conversion!

```
   0 10 1 0 0
× 0+10-1+1-1
```

$$\begin{array}{l}
1111\,111\ \ 01\ 1\ \ 0\ 0 \\
0000\,001\ \ 01\ 0\ \ 0 \\ \hline
0000\,000\ \ 10\ 1\ \ 0\ 0 \\
1111\,101\ \ 10\ 0 \\ \hline
1111\,110\ \ 00\ 1\ \ 0\ 0 \\
0001\,010\ \ 0 \\ \hline
0001\,000\ \ 00\ 1\ \ 0\ 0
\end{array}$$

Text Only:

```
Booth Recoding:
  0 +1  0 -1 +1 -1
Superscript One [¹] = +1
Subscript One [₁] = -1
       010100
     × 0¹0₁¹₁
    ---------
 111111101100
 00000010100
-------------
 000000010100
 1111101100
-------------
 111111000100
 00010100
-------------
 000100000100
```

Booth Multiplier		Bit-Pair Recoding Multiplier	
Bit[i]	Bit[i-1]	Bit[i]	Bit[i-1]
+1	-1	0	+1

-1	+1	0	-1
+1	0	0	+2
-1	0	0	-2
Same as Booth Recoding:			
0	-1	0	-1
0	+1	0	+1
0	0	0	0

Bit-Pair Recoding Table:

Multiplier: 0 0 1 1 0 1 0

Booth Recoding (for Bit-Pair Recoding Method) (Click / Tap Me):	0 +1	0 -1	+1 -1
Bit-Pair:	+1	-1	+1

If the Multiplier is an odd number of bits, a 1/0 bit is added to extent the multiplier to an even number of bits before the most significant bit (MSB) for the Bit-Pair Recoding Method conversion. Since the Multiplier is an even number of bits, we don't add the bit before MSB.

Same as the Booth Recoding above, a red zero is added after the least significant bit (LSB) for the Booth Recoding conversion

Click on the zeros in "Booth Recoding" above to view the pair of bit of each conversion!

```
             0  10 10  0
         ×  0+10-10+1
        ----------------
 00000000  10 10  0
 11111101  10 0
 -----------------
 11111110  00 10  0
 00001010  0
 ----------------
 00001000  00 10  0
```

Text Only:

```
Booth Recoding (for Bit-Pair Recoding Method):
  0 +1  0 -1 +1 -1


Superscript One [¹] = +1
Subscript One [₁] = -1
Superscript Two [²] = +2
Subscript Two [₂] = -2
Bit-Pair Recoding Method:
 +1 -1 +1
        010100
      × 0¹0₁0¹
     ---------
 0000000010100
 11111101100
--------------
 1111111000100
 000010100
--------------
 0000100000100
```

Ques 4. Explain IEEE standard for floating point representation. Represent the number (1460.125)10 in single precision and double precision format.

Ans: The IEEE Standard for Floating-Point Arithmetic (IEEE 754) is a technical standard for floating-point computation which was established in 1985 by the Institute of Electrical and Electronics Engineers (IEEE). The standard addressed many problems found in the diverse floating point implementations that made them difficult to use reliably and reduced their portability. IEEE Standard 754 floating point is the most common representation today for real numbers on computers, including Intel-based PC's, Macs, and most Unix platforms.There are several ways to represent floating point number but IEEE 754 is the most efficient in most cases.

IEEE 754 has 3 basic components:

The Sign of Mantissa –

This is as simple as the name. 0 represents a positive number while 1 represents a negative number.

The Biased exponent –

The exponent field needs to represent both positive and negative exponents. A bias is added to the actual exponent in order to get the stored exponent.

The Normalised Mantissa –

The mantissa is part of a number in scientific notation or a floating-point number, consisting of its significant digits. Here we have only 2 digits, i.e. O and 1. So a normalised mantissa is one with only one 1 to the left of the decimal.

(1460.125)10=(10110110100.001)2

(by using decimal to binary conversion)

Normalizing the number we get:

1.0110110100001 * 210

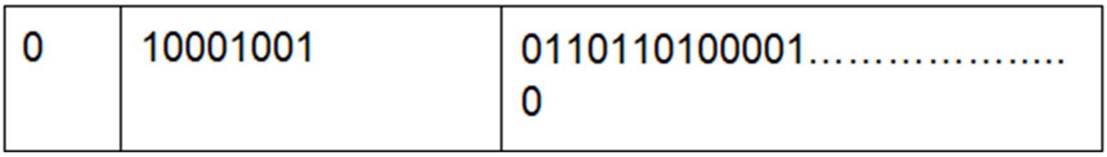
Which gives

M=0110110100001

E=10

E'=127+10=(137)10=(10001001)2

Single Precision Format(32 bits)



0	10001001	0110110100001………………… 0

(1460.125)10=(10110110100.001)2

(by using decimal to binary conversion)

Normalizing the number we get:

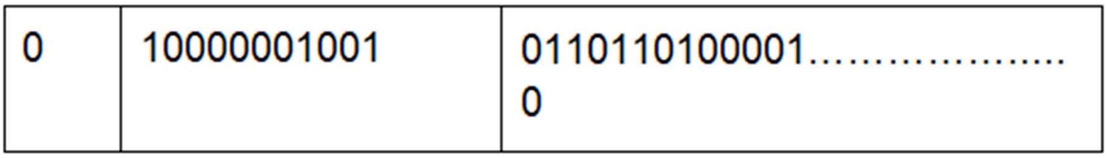
1.0110110100001 * 210

M=0110110100001

E=10

E'=1023+10=(1033)10=(10000001001)2

Double Precision Format(64 bits)



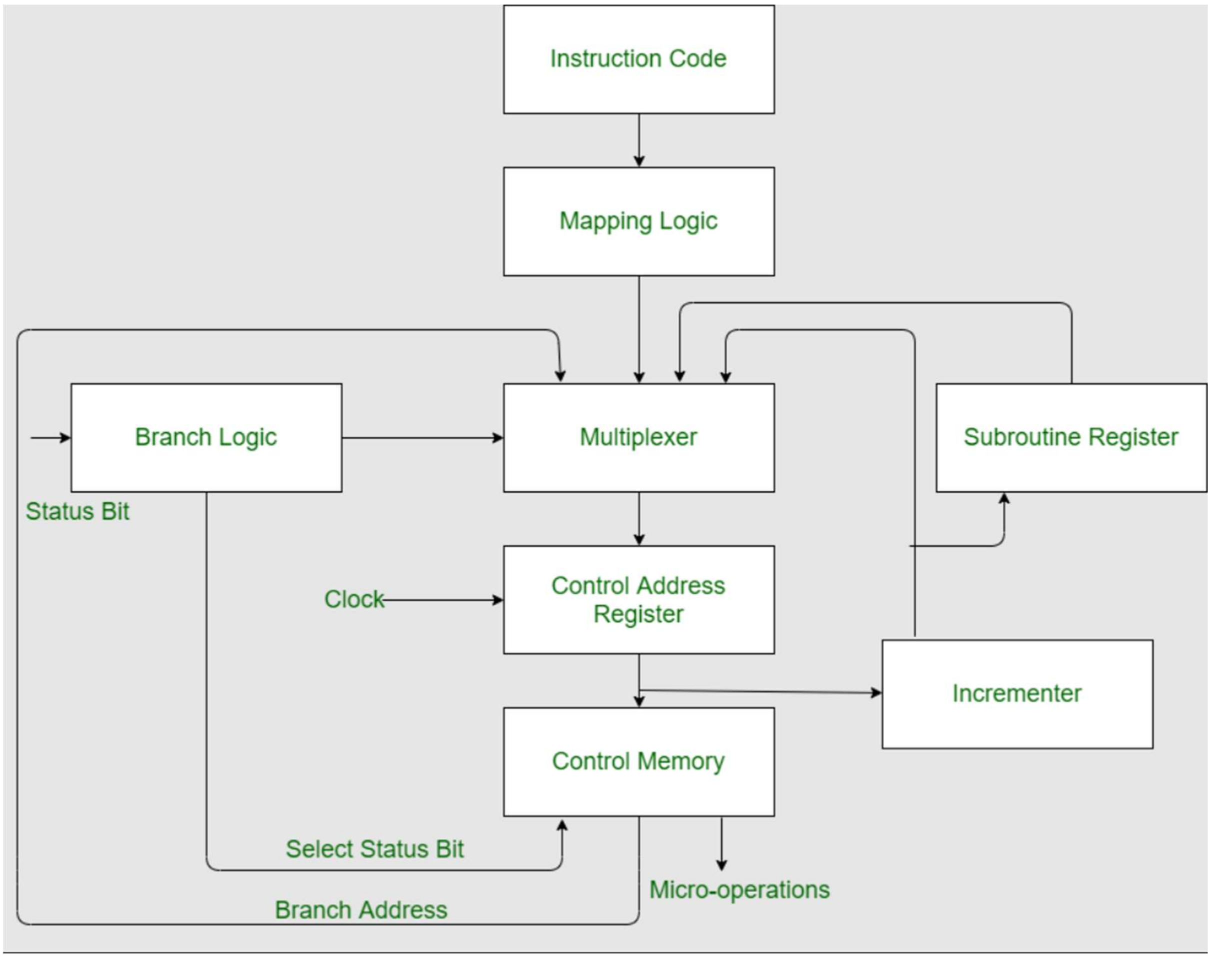
0	10000001001	0110110100001………………… 0

Ques 5.

a) **What is a micro program sequencer? With block diagram, explain the working of micro program sequencer.**

Ans: Micro Instructions Sequencer is a combination of all hardware for selecting the next micro-instruction address. The micro-instruction in control memory contains a set of bits to initiate micro operations in computer registers and other bits to specify the method by which the address is obtained.

Implementation of Micro Instructions Sequencer –



Multiplexer :Multiplexer is a combinational circuit which contains many data inputs and single data output depending on control or select inputs.

Branching :Branching is achieved by specifying the branch address in one of the fields of the micro instruction. Conditional branching is obtained by using part of the micro-instruction to select a specific status bit in order to determine its condition.

Mapping Logic :An external address is transferred into control memory via a mapping logic circuit.

Incrementer :Incrementer increments the content of the control address register by one, to select the next micro-instruction in sequence.

Subroutine Register (SBR) :The return address for a subroutine is stored in a special register called Subroutine Register whose value is then used when the micro-program wishes to return from the subroutine.

Control Memory :Control memory is a type of memory which contains addressable storage registers. Data is temporarily stored in control memory. Control memory can be accessed quicker than main memory.

a) **Differentiate between hardwired and micro programmed control unit. Explain each component of hardwired control unit organization.**

Ans:

Hardwired Control Unit	Microprogrammed Control Unit
Hardwired control unit generates the control signals needed for the processor using logic circuits	Micrprogrammed control unit generates the control signals with the help of micro instructions stored in control memory
Hardwired control unit is faster when compared to microprogrammed control unit as the required control signals are generated with the help of hardwares	This is slower than the other as micro instructions are used for generating signals here
Difficult to modify as the control signals that need to be generated are hard wired	Easy to modify as the modification need to be done only at the instruction level
More costlier as everything has to be realized in terms of logic gates	Less costlier than hardwired control as only micro instructions are used for generating control signals
It cannot handle complex instructions as the circuit design for it becomes complex	It can handle complex instructions
Only limited number of instructions are used due to the hardware implementation	Control signals for many instructions can be generated
Used in computer that makes use of Reduced Instruction Set Computers(RISC)	Used in computer that makes use of Complex Instruction Set Computers(CISC)

Hardwired Control

a) The Hardwired Control organization involves the control logic to be implemented with gates, flip-flops, decoders, and other digital circuits.
b) The following image shows the block diagram of a Hardwired Control organization.

c) A Hard-wired Control consists of two decoders, a sequence counter, and a number of logic gates.

d) An instruction fetched from the memory unit is placed in the instruction register (IR).

e) The component of an instruction register includes; I bit, the operation code, and bits 0 through 11.

f) The operation code in bits 12 through 14 are coded with a 3 x 8 decoder.

g) The outputs of the decoder are designated by the symbols D0 through D7.

h) The operation code at bit 15 is transferred to a flip-flop designated by the symbol I.

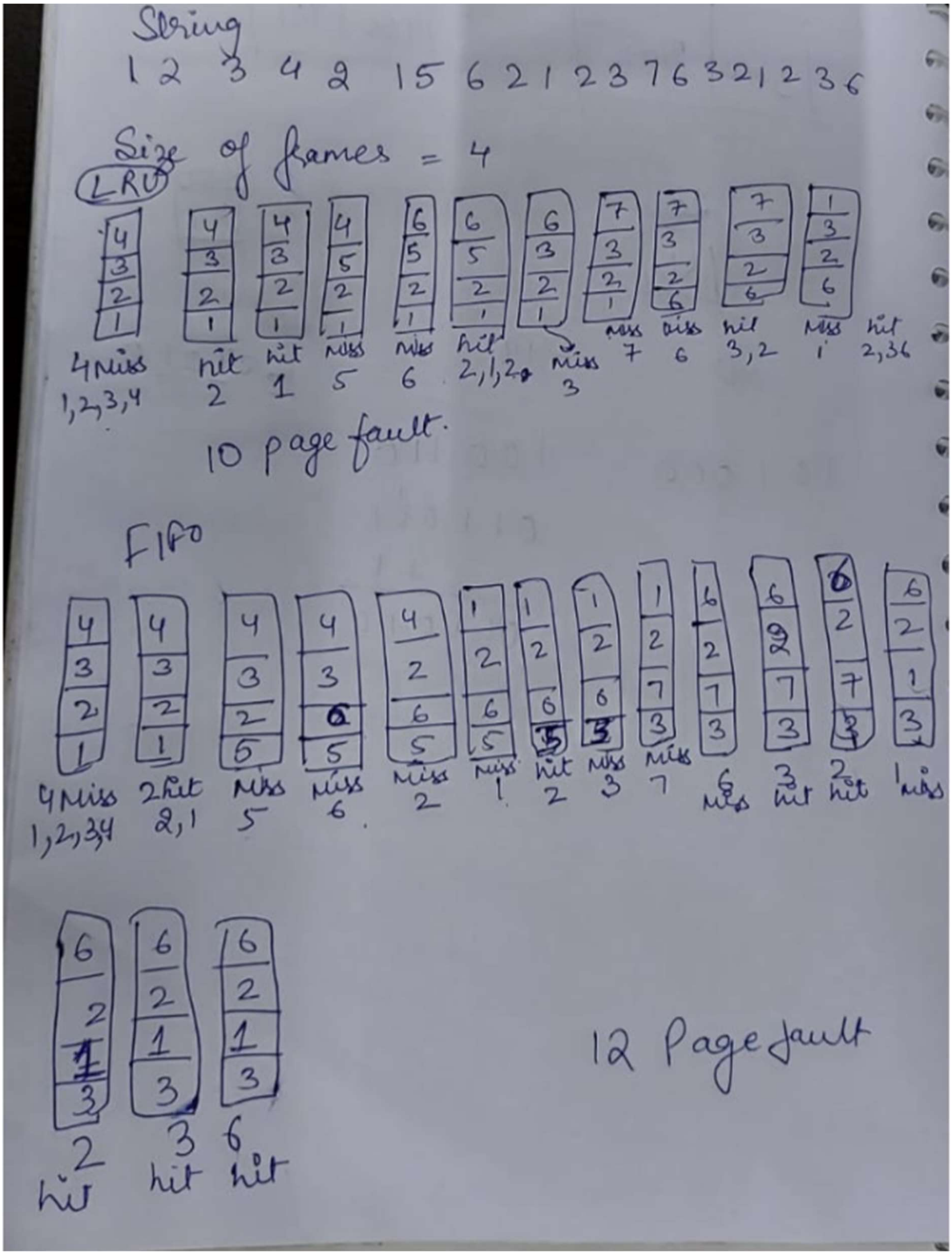
i) The operation codes from Bits 0 through 11 are applied to the control logic gates.

j) The Sequence counter (SC) can count in binary from 0 through 15.

Ques 6.

a) Calculate the page fault for a given string with the help of LRU & FIFO page replacement algorithm, Size of frames = 4 and string 1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6.

Ans:



String

1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6

Size of frames = 4

LRU

4	4	4	4	6	6	6	7	7	7	1	
3	3	3	5	5	5	3	3	3	3	3	
2	2	2	2	2	2	2	2	2	2	2	
1	1	1	1	1	1	1	1	6	6	6	
4 Miss 1,2,3,4	hit 2	hit 1	Miss 5	Miss 6	hit 2,1,2	Miss 3	Miss 7	Miss 6	hit 3,2	Miss 1	hit 2,36

10 page fault.

FIFO

4	4	4	4	4	1	1	1	1	6	6	6	6	6	6	6
3	3	3	3	2	2	2	2	2	2	2	2	2	2	2	2
2	2	2	6	6	6	6	6	7	7	7	7	1	1	1	1
1	1	5	5	5	5	5	3	3	3	3	3	3	3	3	3
4 Miss 1,2,3,4	2 hit 2,1	Miss 5	Miss 6	Miss 2	Miss 1	hit 2	Miss 3	Miss 7	6 Miss	3 hit	2 hit	1 miss	2 hit	3 hit	6 hit

12 Page fault

b) A computer uses RAM chips of 1024* 1 capacity.

i) How many chips are needed & how should their address lines be connected to provide a memory capacity of 1024*8?

ii) How many chips are needed to provide a memory capacity of 16 KB?

Ans: (i) As given available chips = 1024 x 1 capacity and

Required capacity = 1024 x 8 capacity

Number of Chips= (1024X8)/(1024X1) = 8

Number of address lines are needed = 10 (that is 1024 = 210)

Here the word capacity is similar (1024) so similar address lines will be connected to all chips.

(ii) Number Of Chips Required = (16X1024X8)/(1024X1) = 128

Ques 7.

a) **What do you mean by asynchronous data transfer? Explain strobe control and hand shaking mechanism.**

Ans: Asynchronous Transmission

In asynchronous transmission, data moves in a half-paired approach, 1 byte or 1 character at a time. It sends the data in a constant current of bytes. The size of a character transmitted is 8 bits, with a parity bit added both at the beginning and at the end, making it a total of 10 bits. It doesn't need a clock for integration—rather, it utilises the parity bits to tell the receiver how to translate the data.

It is straightforward, quick, cost-effective, and doesn't need 2-way communication to function.

Characteristics of Asynchronous Transmission

a) Each character is headed by a beginning bit and concluded with one or more end bits.
b) There may be gaps or spaces in between characters.
c) Examples of Asynchronous Transmission

d) Emails
e) Forums
f) Letters
g) Radios
h) Televisions

Asynchronous input output is a form of input output processing that allows others devices to do processing before the transmission or data transfer is done.

Problem faced in asynchronous input output synchronization –

It is not sure that the data on the data bus is fresh or not as their no time slot for sending or receiving data.

This problem is solved by following mechanism:

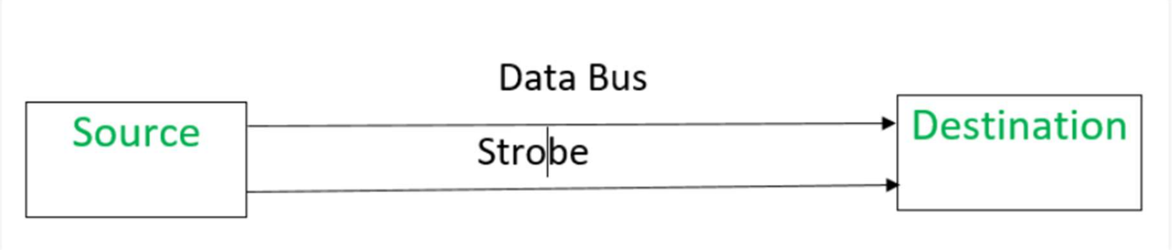
a) Strobe
b) Handshaking

Data is transferred from source to destination through data bus in between.

1. Strobe Mechanism:

Source initiated Strobe – When source initiates the process of data transfer. Strobe is just a signal.

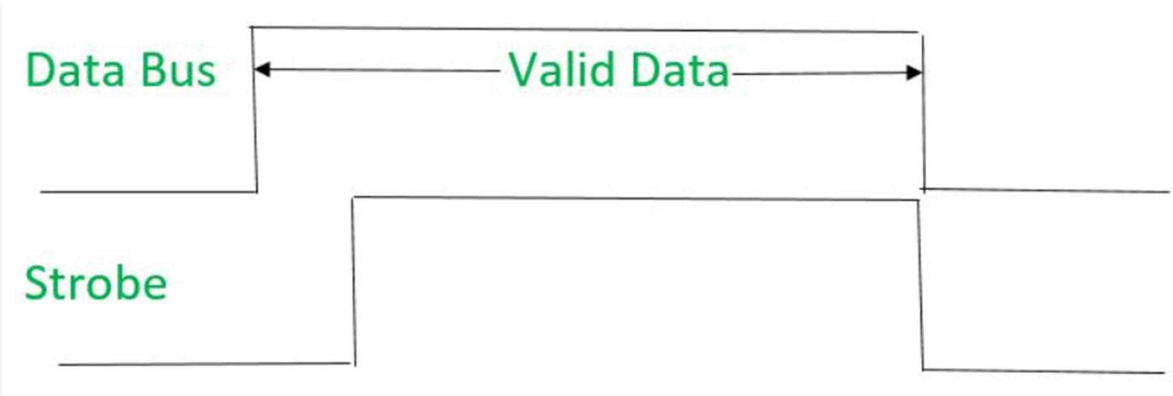
(i) First, source puts data on the data bus and ON the strobe signal.



(ii) Destination on seeing the ON signal of strobe, read data from the data bus.

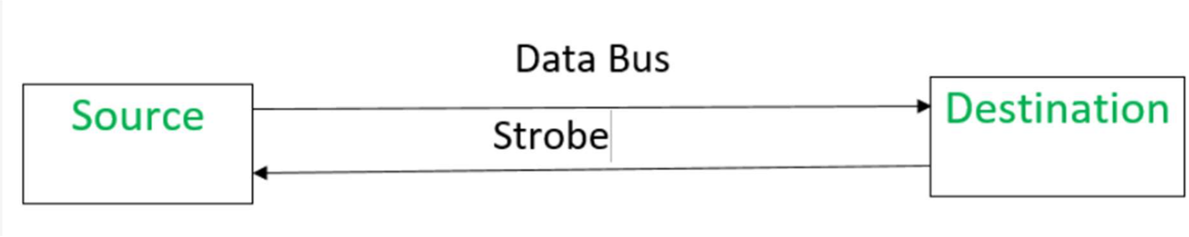
(iii) After reading data from the data bus by destination, strobe gets OFF.

Signals can be seen as:



It shows that first data is put on the data bus and then strobe signal gets active.

Destination initiated signal – When destination initiates the process of data transfer.

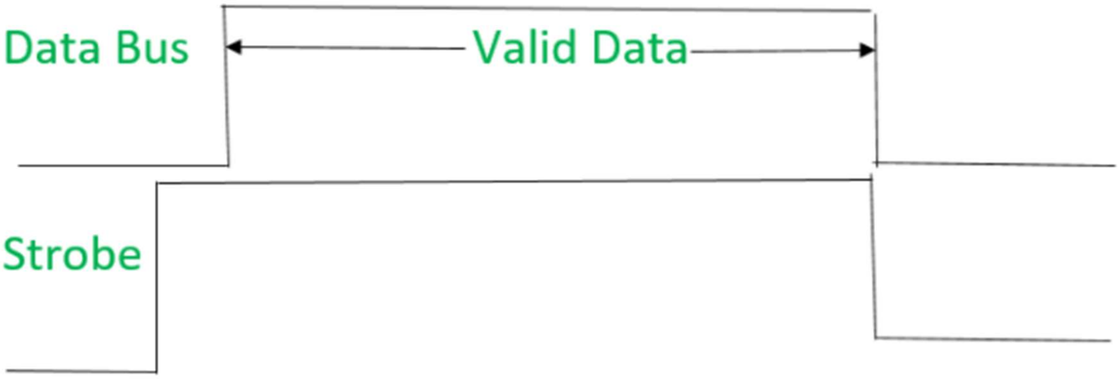


(i) First, the destination ON the strobe signal to ensure the source to put the fresh data on the data bus.

(ii) Source on seeing the ON signal puts fresh data on the data bus.

(iii) Destination reads the data from the data bus and strobe gets OFF signal.

Signals can be seen as:



It shows that first strobe signal gets active then data is put on the data bus.

Problems faced in Strobe based asynchronous input output –

a) In Source initiated Strobe, it is assumed that destination has read the data from the data bus but their is no surety.
b) In Destination initiated Strobe, it is assumed that source has put the data on the data bus but their is no surety.

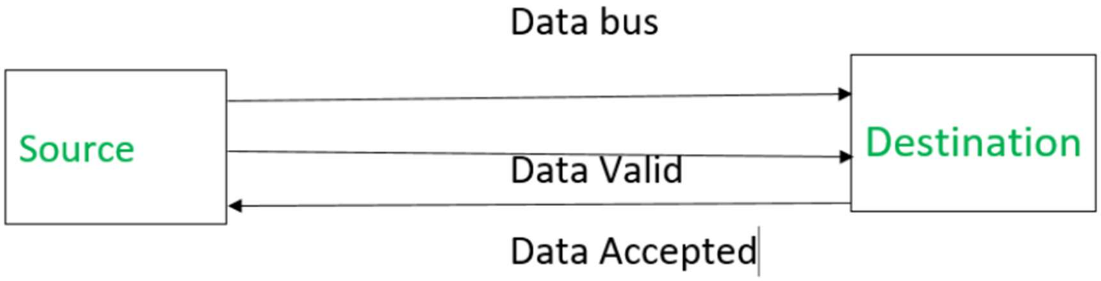
This problem is overcome by Handshaking.

2. Handshaking Mechanism:

Source initiated Handshaking – When source initiates the data transfer process. It consists of signals:

DATA VALID: if ON tells data on the data bus is valid otherwise invalid.

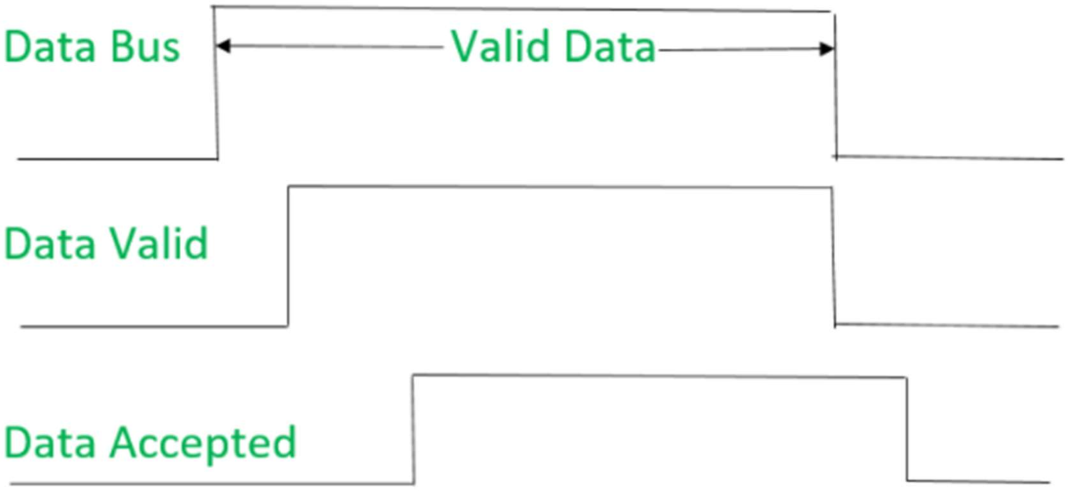
DATA ACCEPTED: if ON tells data is accepted otherwise not accepted.



(i) Source places data on the data bus and enable Data valid signal.
(ii) Destination accepts data from the data bus and enable Data accepted signal.
(iii) After this, disable Data valid signal means data on data bus is invalid now.
(iv) Disable Data accepted signal and the process ends.

Now there is surety that destination has read the data from the data bus through data accepted signal.

Signals can be seen as:

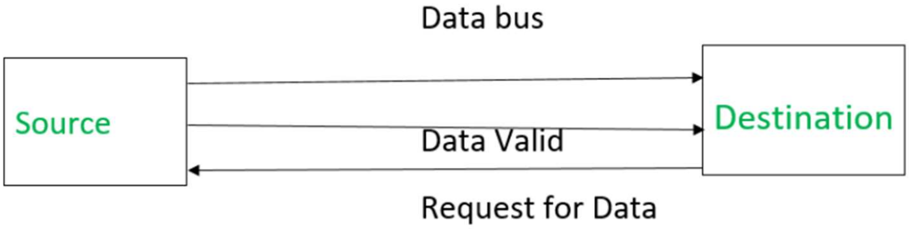


It shows that first data is put on the data bus then data valid signal gets active and then data accepted signal gets active. After accepting the data, first data valid signal gets off then data accepted signal gets off

Destination initiated Handshaking – When destination initiates the process of data transfer.

REQUEST FOR DATA: if ON requests for putting data on the data bus.

DATA VALID: if ON tells data is valid on the data bus otherwise invalid data.



(i) When destination is ready to receive data, Request for Data signal gets activated.

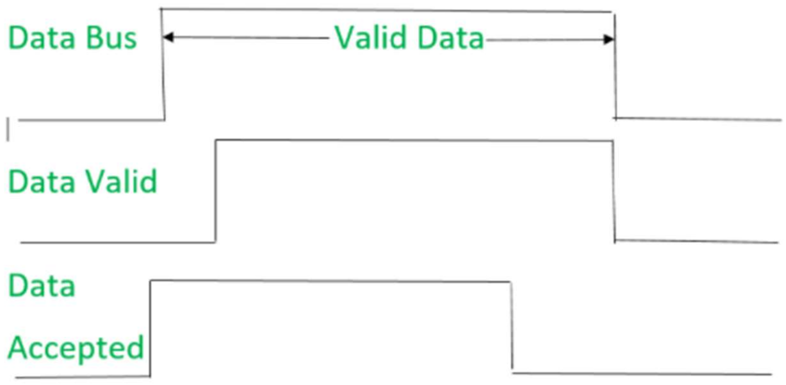
(ii) source in response puts data on the data bus and enabled Data valid signal.

(iii) Destination then accepts data from the data bus and after accepting data, disabled Request for Data signal.

(iv) At last, Data valid signal gets disabled means data on the data bus is no more valid data.

Now there is surety that source has put the data on the data bus through data valid signal.

Signals can be seen as:



It shows that first Request for Data signal gets active then data is put on data bus then Data valid signal gets active. After reading data, first Request for Data signal gets off then Data valid signal.

i) Discuss the different modes of data transfer.

Ans: The method that is used to transfer information between internal storage and external I/O devices is known as I/O interface. The CPU is interfaced using special communication links by the peripherals connected to any computer system. These communication links are used to resolve the differences between CPU and peripheral. There exists special hardware components between CPU and peripherals to supervise and synchronize all the input and output transfers that are called interface units.

Mode of Transfer:

The binary information that is received from an external device is usually stored in the memory unit. The information that is transferred from the CPU to the external device is originated from the memory unit. CPU merely processes the information but the source and target is always the memory unit. Data transfer between CPU and the I/O devices may be done in different modes.

Data transfer to and from the peripherals may be done in any of the three possible ways

Programmed I/O.

Interrupt- initiated I/O.

Direct memory access(DMA).

Programmed I/O: It is due to the result of the I/O instructions that are written in the computer program. Each data item transfer is initiated by an instruction in the program. Usually the transfer is from a CPU register and memory. In this case it requires constant monitoring by the CPU of the peripheral devices.

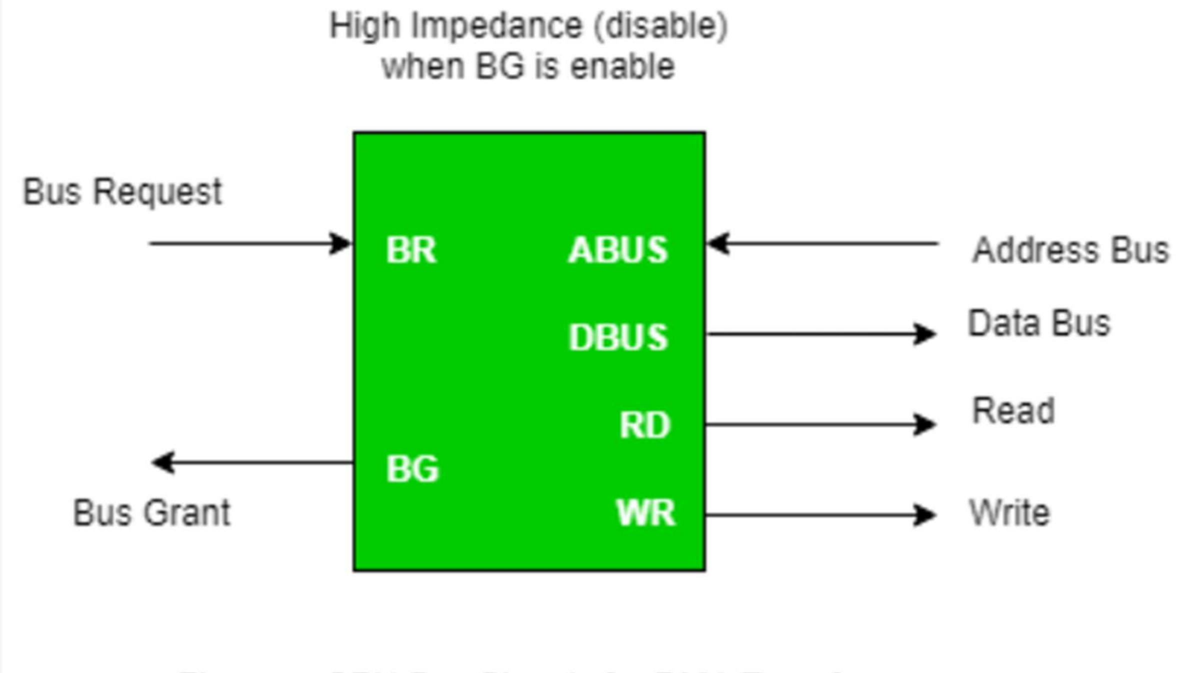


Figure - CPU Bus Signals for DMA Transfer

Example of Programmed I/O: In this case, the I/O device does not have direct access to the memory unit. A transfer from I/O device to memory requires the execution of several instructions by the CPU, including an input instruction to transfer the data from device to the CPU and store instruction to transfer the data from CPU to memory. In programmed I/O, the CPU stays in the program loop until the I/O unit indicates that it is ready for data transfer.

This is a time consuming process since it needlessly keeps the CPU busy. This situation can be avoided by using an interrupt facility. This is discussed below.

Interrupt- initiated I/O: Since in the above case we saw the CPU is kept busy unnecessarily. This situation can very well be avoided by using an interrupt driven method for data transfer. By using interrupt facility and special commands to inform the interface to issue an interrupt request signal whenever data is available from any device. In the meantime the CPU can proceed for any other program execution. The interface meanwhile keeps monitoring the device. Whenever it is determined that the device is ready for data transfer it initiates an interrupt request signal to the computer. Upon detection of an external interrupt signal the CPU stops momentarily the task that it was already performing, branches to the service program to process the I/O transfer, and then return to the task it was originally performing.

Note: Both the methods programmed I/O and Interrupt-driven I/O require the active intervention of the

processor to transfer data between memory and the I/O module, and any data transfer must transverse

a path through the processor. Thus both these forms of I/O suffer from two inherent drawbacks.

The I/O transfer rate is limited by the speed with which the processor can test and service a device.

The processor is tied up in managing an I/O transfer; a number of instructions must be executed for each I/O transfer.

Direct Memory Access: The data transfer between a fast storage media such as magnetic disk and memory unit is limited by the speed of the CPU. Thus we can allow the peripherals directly communicate with each other using the memory buses, removing the intervention of the CPU. This type of data transfer technique is known as DMA or direct memory access. During DMA the CPU is idle and it has no control over the memory buses. The DMA controller takes over the buses to manage the transfer directly between the I/O devices and the memory unit.

Bus Request : It is used by the DMA controller to request the CPU to relinquish the control of the buses.

Bus Grant : It is activated by the CPU to Inform the external DMA controller that the buses are in high impedance state and the requesting DMA can take control of the buses. Once the DMA has taken the control of the buses it transfers the data. This transfer can take place in many ways.

Types of DMA transfer using DMA controller:

Burst Transfer :

DMA returns the bus after complete data transfer. A register is used as a byte count,

being decremented for each byte transfer, and upon the byte count reaching zero, the DMAC will

release the bus. When the DMAC operates in burst mode, the CPU is halted for the duration of the data

transfer.

Steps involved are:

Bus grant request time.

Transfer the entire block of data at transfer rate of device because the device is usually slow than the

speed at which the data can be transferred to CPU.

Release the control of the bus back to CPU

So, total time taken to transfer the N bytes

= Bus grant request time + (N) * (memory transfer rate) + Bus release control time.

Where,

X µsec =data transfer time or preparation time (words/block)

Y µsec =memory cycle time or cycle time or transfer time (words/block)

% CPU idle (Blocked)=(Y/X+Y)*100

% CPU Busy=(X/X+Y)*100

Cyclic Stealing :

An alternative method in which DMA controller transfers one word at a time after which it must return the control of the buses to the CPU. The CPU delays its operation only for one memory cycle to allow the direct memory I/O transfer to “steal” one memory cycle.

Steps Involved are:

1. Buffer the byte into the buffer
2. Inform the CPU that the device has 1 byte to transfer (i.e. bus grant request)
3. Transfer the byte (at system bus speed)
4. Release the control of the bus back to CPU.

Before moving on transfer next byte of data, device performs step 1 again so that bus isn’t tied up and

the transfer won’t depend upon the transfer rate of device.

So, for 1 byte of transfer of data, time taken by using cycle stealing mode (T).

= time required for bus grant + 1 bus cycle to transfer data + time required to release the bus, it will be

N x T

In cycle stealing mode we always follow pipelining concept that when one byte is getting transferred then Device is parallel preparing the next byte. “The fraction of CPU time to the data transfer time” if asked then cycle stealing mode is used.

Where,

X µsec =data transfer time or preparation time

(words/block)

Y µsec =memory cycle time or cycle time or transfer

time (words/block)

% CPU idle (Blocked) =(Y/X)*100

% CPU busy=(X/Y)*100

Interleaved mode: In this technique , the DMA controller takes over the system bus when the microprocessor is not using it.An alternate half cycle i.e. half cycle DMA + half cycle processor.